LAPORAN TUGAS 4

Object, Class dan Ecapsulation pada Java

Laporan ini disusun untuk memenuhi Tugas Mata Kuliah Praktikum Teknik Pemograman

Disusun oleh:
Reka Briyan Cahya Heryana 211524024

PROGRAM STUDI D4 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG
2021

KASUS 1

1. Hasil Program

```java
*/
package kasus1;

public class Barang {
String kode_barang;
String nama_barang;
/*
stok akan dienkapsulasi dengan menggunakan
variabel privat untuk melarang penggunaan
operasi matematika lain selain tambah
*/

private int stok;


public int getStok(){
    return stok; //untuk mendapatkan stok setelah menambahkan kuantitas
}

public void setStok(int stok){
    this.stok += stok; //untuk mengatur stok dengan menambahkan kuantitas
}

public Barang(String kode, String nama, int stk) {
kode_barang = kode;
nama_barang = nama;
stok = stk;
}
```

Gambar 1 : Screenshoot program Class Barang

```java
public class Inventori {
Barang[] barangs;

void initBarang() {
barangs = new Barang[2]; //membuat dua array untuk menampung dua barang
barangs[0] = new Barang("001", "Baju", 10); // barang pertama
barangs[1] = new Barang("002", "Celana", 20); // barang kedua
}
void showBarang() {
    /*
    untuk menampilkan barang 1 dan barang 2 serta menampilkan kuantitas
    stok baru setelah menambahkan dengan stok baru di setStok
    */
System.out.println(barangs[0].nama_barang + "(" + barangs[0].getStok() + ")");
System.out.println(barangs[1].nama_barang + "(" + barangs[1].getStok() + ")");
}
void pengadaan() {
initBarang();
/*menambah stocks*/
barangs[0].setStok(10); /* menambah 10 stock*/
barangs[0].setStok(20); /* menambah 20 stock*/
barangs[0].setStok(25); /* menambah 25 stock*/
showBarang(); /* menampilkan nomor dan nama setelah menambahkan stock baru*/
}

public static void main(String[] args) {
        Inventori beli = new Inventori(); // untuk memanggil class inventori
        beli.pengadaan(); // untuk memanggil void pengadaan dan penambahan stok
```

Gambar 2 : Screenshoot program Class Inventori

2. Jawaban yang Dipertanyakan

```
run:
Baju(65)
Celana(20)
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 3 : Screenshoot output dari program

3. Permasalahan yang Ditemukan
   - Belum mengerti dan bingung dalam mengaplikasikan maksud dari *encapsulation*
4. Solusi dari Permasalahan yang Dihadapi
   - Pada kasus ini saya mencari informasi di internet tentang *encapsulation.* Pada kasus ini juga saya memecahkan masalah ini dengan cara melakukan *encapsulation* pada *int stok* sehingga untuk melakukan pengadaan barang dan penambahan stok tidak dapat langsung mengaksek pada *stok* atau kita perlu sebuah *method* khusus yang dapat melakukan hal ini. Maka dari itu dengan melakukan encapsulation ini kita tidak dapat memberikan peritntah aritmatika seperti pada gambar dibawah.

```
barangs[0].stok += 20;
barangs[0].stok -= 30; //Bisa juga dikurangi dong?
barangs[0].stok *= 30; //dikali juga bisa dong??
```

5. Teman yang membantu
   - Tidak ada

KASUS 2

1. Hasil Program

```java
/*
 Pada class ini berisikan metode untuk mencetak nama item
yang akan digunakan pada class UpinIpin
 */
package kasus2;

public class Item {
    // Kedua variabel ini diatur sebagai private untuk enkapsulasi variabel

    private String name;

    private Item() {
    name = "Ipin";
}
    public Item(String name) {
    this.name = new Item().name;
    System.out.println(this.name);
}
}
```

Gambar 1 : Screenshoot program Class Item

```java
/*
 Class ini merupakan Class inti
Class ini juga menggunakan Kelas item untuk mencetak
string yang telah dinyatakan sebelum Kelas item.
 */
package kasus2;

public class UpinIpin {
    public static void main(String[] args) {
    Item name = new Item("upin");
}
}
```

Gambar 2 : Screenshoot program Class UpinIpin

2. Jawaban yang Dipertanyakan

```
run:
Ipin
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 3 : Screenshoot hasil output dari Program

3. Permasalahan yang Ditemukan
   - Tidak tahu apa itu construct dan fungsi this() pada java
4. Solusi dari Permasalahan yang Dihadapi
   - Mencari informasi di internet dan mencoba mengaplikasikannya. Untuk menghasilkan sebuah output “Ipin” ada dua cara diantaranya yang pertama dengan menambahkan fungsi **this()**, seperti pada gambar 4. Yang kedua dengan cara mengisi **data private**

String name sama dengan “Ipin”. Fungsi **this()** ini akan memanggil **private Item()** sehingga data **name** sama dengan “Ipin”.

```
public Item(String name) {
this();
System.out.println(this.name);
```

Gambar 4

5. Teman yang membantu
   - Tidak ada

KASUS 3

1. Hasil Program

```
/*
 Class ini mencetak nomor dari Classnya sendiri
dan akan dipanggil oleh Class main sebagai tipe yang baru
 */
package kasus3;

public class KelasSatu {
    {
System.out.println(11);
/* akan muncul setiap kali setelah satu nomor
dari Class KelasSatu muncul yang bertindak
sebagai dari inisialisasi block statis*/
}
static
{
System.out.println(2); // akan muncul satu kali di awal ketika construct dipanggil
}
public KelasSatu(int i)
{
System.out.println(3); // akan muncul ketika constructor KelasSatu dengan parameter dipanggil
}
public KelasSatu()
{
System.out.println(4); // akan muncul ketika constructor KelasSatu dipanggil dan tidak memiliki parameter
}
}
```

Gambar 1 : Screenshoot program Class KelasSatu

```
/*
 Class ini akan menampilkan nomor pada classnya sendiri
dan nomor dari Class KelasSatu yang dipanggil menggunakan conctract
 */
package kasus3;

public class KelasDua {
    {
System.out.println(5);
/* akan muncul pada urutan terakhir
dan hanya dieksekusi jika Class KelasDua mendapatkan contructed*/
}
public static void main(String[] args){
System.out.println(6); // akan muncul pada urutan pertama
KelasSatu satu = new KelasSatu();// contruct object tanpa parameter
KelasSatu dua = new KelasSatu(10);// contruct object dengan parameter
}
}
```

Gambar 2 : Screenshoot program Class KelasDua

2. Jawaban yang Dipertanyakan

```
run:
6
2
11
4
11
3
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Permasalahan yang Ditemukan
   - Tidak mengetahui *Static Block* yang terdapat pada Class KelasSatu
   - Tidak mengetahui initializer yang terdapat pada Class KelasDua
4. Solusi dari Permasalahan yang Dihadapi
   - Mencari informasi di internet. *Static Blocks* yang terdapat pada Class KelasSatu merupakan special blocks di Java. *Static Blocks* hanya akan dijalankan sekali saja dan

waktu pertama kali. Sedangkan initializer yang terdapat pada Class KelasDua dan memiliki arti blocks yang tidak memiliki identitas serta hanya akan tampil ketika dibuat objek dari class tersebut.

- Baris pertama dari kasus ini akan menampilkan angka 6
- Baris kedua dari kasus ini akan menampilkan angka 2 yang berasal dari *static blocks*
- Baris ketiga dari kasus ini akan menampilkan angka 11 yang berasal dari *blocks* sebelum *static blocks*
- Baris keempat dari kasus ini akan menampilkan angka 4 dari *constructor* KelasSatu yang tidak memiliki parameter.
- Baris kelima akan menampilkan angka 11 yang berasal dari *blocks* sebelum *static blocks*
- Baris keenam atau yang terakhir akan menampilkan angka 3 yang berasal dari *constructor* KelasSatu yang memiliki parameter, namun parameter ini tidak digunakan untuk menampilkan ke layar maka dari itu angka 10 tidak akan tampil.

5. Teman yang membantu
   - Tidak ada